```
EEEEEEEEEEEEEEE  DDDDDDDDDDD     FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFFFFFFFFFFFFFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFFFFFFFFFFFFFF
EEE              DDD         DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEEEEEEEEEEE     DDD          DDD FFFFFFFFFFFF
EEEEEEEEEEEE     DDD          DDD FFFFFFFFFFFF
EEEEEEEEEEEE     DDD          DDD FFFFFFFFFFFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEE              DDD          DDD FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDD     FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
EEEEEEEEEEEEEEE  DDDDDDDDDDDD    FFF
```

```
_$
Va
--
00
00
00
00
00
00
00
00
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
7F
```

```
EEEEEEEEEE  DDDDDDDD    FFFFFFFFFF  VV      VV    AAAAAA    LL          UU      UU  EEEEEEEEEE
EEEEEEEEEE  DDDDDDDD    FFFFFFFFFF  VV      VV    AAAAAA    LL          UU      UU  EEEEEEEEEE
EE          DD      DD  FF          VV      VV  AA      AA  LL          UU      UU  EE
EE          DD      DD  FF          VV      VV  AA      AA  LL          UU      UU  EE
EE          DD      DD  FF          VV      VV  AA      AA  LL          UU      UU  EE
EE          DD      DD  FF          VV      VV  AA      AA  LL          UU      UU  EE
EEEEEEEE    DD      DD  FFFFFFFF    VV      VV  AA      AA  LL          UU      UU  EEEEEEEE
EEEEEEEE    DD      DD  FFFFFFFF    VV      VV  AA      AA  LL          UU      UU  EEEEEEEE
EE          DD      DD  FF          VV      VV  AAAAAAAAAA  LL          UU      UU  EE
EE          DD      DD  FF          VV      VV  AAAAAAAAAA  LL          UU      UU  EE
EE          DD      DD  FF            VV  VV    AA      AA  LL          UU      UU  EE                ....
EE          DD      DD  FF            VV  VV    AA      AA  LL          UU      UU  EE                ....
EEEEEEEEEE  DDDDDDDD    FF              VV      AA      AA  LLLLLLLLLL  UUUUUUUUUU  EEEEEEEEEE        ....
EEEEEEEEEE  DDDDDDDD    FF              VV      AA      AA  LLLLLLLLLL  UUUUUUUUUU  EEEEEEEEEE        ....
```

```
PPPPPPPP      AAAAAA      SSSSSSSS
PPPPPPPP      AAAAAA      SSSSSSSS
PP      PP  AA      AA  SS
PP      PP  AA      AA  SS
PP      PP  AA      AA  SS
PP      PP  AA      AA  SS
PPPPPPPP    AA      AA    SSSSSS
PPPPPPPP    AA      AA    SSSSSS
PP          AAAAAAAAAA          SS
PP          AAAAAAAAAA          SS
PP          AA      AA          SS
PP          AA      AA          SS
PP          AA      AA  SSSSSSSS
PP          AA      AA  SSSSSSSS
```

```
{ ++

            FILE: SRC$:EDFVALUE.PAS - Pascal include file to define
            initial values of selected top-level variables.
```



```
FACILITY:       VAX/VMS EDF (EDIT/FDL) UTILITY

ABSTRACT:       This facility is used to create, modify, and optimize
                FDL specification files.

ENVIRONMENT:    NATIVE/USER MODE

AUTHOR:         Ken F. Henderson Jr.

CREATION DATE:  27-Mar-1981

MODIFIED BY:

                V03-009 KFH0009         Ken Henderson           10 Sep 1983
                        Support named UICs.

                V03-008 KFH0008         Ken Henderson            9 Aug 1983
                        Fix max value of CLUSTER_SIZE.
                        Fix default of QTAB[TEST_PRIMARY].

                V03-007 KFH0007         Ken Henderson           30 Jul 1983
                        Fix SEC_TYPE table for audit_trail.
                        Add DEFERRED_WRITE.
```

```
                V03-006 KFH0006         Ken Henderson           26 Apr 1983
                        Fix various defaults in QTAB.
                        Transferred some initialization to
                        the EDFVAR declarations.

                V03-005 KFH0005         Ken Henderson           14 Apr 1983
                        Changed max bucketsize to 63 from 65.
                        Added ANALYSIS, OUTPUT, RESPONSES,
                        PROMPTING, SET_FUNCTION, GRANULARITY.
                        Added support for SEGMENTED keys.

                V03-004 KFH0004         Ken Henderson            7 Mar 1983
                        Changed max bucketsize to 65 from 127.

                V03-003 KFH0003         Ken Henderson           11 Sept 1982
                        Added initialization of VDATA and BDATA.

                V03-002 KFH0002         Ken Henderson           9 Sept 1982
                        Added initialization of QTAB.

                V03-001 KFH0001         Ken Henderson           23-Mar-1982
                        Took out reference to EDITFDL_STRING

-- }
```

```
    { +
    Initialize the Boolean-array to all false.
    - }
    BDATA       := (
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE
                );

    { +
    Initialize the String-array to all null-string.
    - }
    SDATA       := (

        (0,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
        (0,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
        (0,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
        (0,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
        (0,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
        (0,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL)

                );

    { +
    Initialize the Valid-array to all false.
    - }
    VDATA       := (
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
                FALSE,
```

```
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE
```

```
                );

    { +
    Initialize the sequencing array.
    - }
    PRI_SEQ     := (
                    15,         { DUMMY_PRIMARY$ }

                    8,          { ACCESS, }
                    4,          { ACL, }
                    13,         { ANALYSIS_OF_AREA, }
                    14,         { ANALYSIS_OF_KEY, }
                    11,         { AREA, }
                    10,         { CONNECT, }
                    4,          { DATE, }
                    3,          { FILE$, }
                    1,          { IDENT, }
                    6,          { JOURNAL, }
                    12,         { KEY, }
                    7,          { RECORD$, }
                    9,          { SHARING, }
                    2,          { SYSTEM, }
                    0           { TITLE }
                    );

    { +
    Initialize the 'width' arrays - that indicate how long a particular
    keyword should be printed.
    - }
    PRIMARY_WIDTH := (
                    0,          { DUMMY_PRIMARY$ }

                    6,          { ACCESS, }
                    3,          { ACL, }
                    16,         { ANALYSIS_OF_AREA, }
                    15,         { ANALYSIS_OF_KEY, }
                    4,          { AREA, }
                    7,          { CONNECT, }
                    4,          { DATE, }
                    4,          { FILE$, }
                    5,          { IDENT, }
                    7,          { JOURNAL, }
                    3,          { KEY, }
                    6,          { RECORD$, }
                    7,          { SHARING, }
                    6,          { SYSTEM, }
                    5           { TITLE }
                    );

    SECONDARY_WIDTH := (

{ RESERVE 0 }       0,          { DUMMY_SECONDARY$, }

{ ACCESS PRIMARY }

                    8,          { BLOCK_IO$ }
```

```
                    6,          { DELETE$ }
                    3,          { GET$ }
                    3,          { PUT$ }
                    9,          { RECORD_IO$ }
                    8,          { TRUNCATE$ }
                    6,          { UPDATE$ }
{ ACL PRIMARY }
                    5,          { ENTRY }
{ ANALYSIS_OF_AREA PRIMARY }
                    15,         { RECLAIMED_SPACE }
{ ANALYSIS_OF_KEY PRIMARY }
                    9,          { DATA_FILL$, }
                    20,         { DATA_KEY_COMPRESSION, }
                    23,         { DATA_RECORD_COMPRESSION, }
                    17,         { DATA_RECORD_COUNT, }
                    19,         { DATA_SPACE_OCCUPIED, }
                    9,          { DELETIONS, }
                    5,          { DEPTH, }
                    19,         { DUPLICATES_PER_SIDR, }
                    17,         { INDEX_COMPRESSION, }
                    10,         { INDEX_FILL$, }
                    20,         { INDEX_SPACE_OCCUPIED, }
                    19,         { LEVEL1_RECORD_COUNT }
                    16,         { MEAN_DATA_LENGTH, }
                    17,         { MEAN_INDEX_LENGTH, }
                    15,         { RANDOM_ACCESSES, }
                    14,         { RANDOM_INSERTS, }
                    19,         { SEQUENTIAL_ACCESSES, }
{ AREA PRIMARY }
                    10,         { ALLOCATION$, }
                    19,         { BEST_TRY_CONTIGUOUS$, }
                    11,         { BUCKET_SIZE$, }
                    10,         { CONTIGUOUS$, }
                    17,         { EXACT_POSITIONING$, }
                    9,          { EXTENSION$, }
                    8,          { POSITION$, }
                    6,          { VOLUME$, }
{ CONNECT PRIMARY }
                    12,         { ASYNCHRONOUS }
                    8,          { BLOCK_IO }
                    11,         { BUCKET_CODE }
                    7,          { CONTEXT }
                    11,         { END_OF_FILE }
                    12,         { FILL_BUCKETS }
                    11,         { FAST_DELETE }
                    16,         { KEY_OF_REFERENCE }
                    17,         { KEY_GREATER_EQUAL }
```

```
                16,         { KEY_GREATER_THAN }
                9,          { KEY_LIMIT }
                11,         { LOCATE_MODE }
                12,         { LOCK_ON_READ }
                13,         { LOCK_ON_WRITE }
                16,         { MANUAL_UNLOCKING }
                16,         { MULTIBLOCK_COUNT }
                17,         { MULTIBUFFER_COUNT }
                6,          { NOLOCK }
                18,         { NONEXISTENT_RECORD }
                10,         { READ_AHEAD }
                15,         { READ_REGARDLESS }
                14,         { TIMEOUT_ENABLE }
                14,         { TIMEOUT_PERIOD }
                15,         { TRUNCATE_ON_PUT }
                19,         { TT_CANCEL_CONTROL_O }
                15,         { TT_UPCASE_INPUT }
                9,          { TT_PROMPT }
                19,         { TT_PURGE_TYPE_AHEAD }
                14,         { TT_READ_NOECHO }
                16,         { TT_READ_NOFILTER }
                9,          { UPDATE_IF }
                15,         { WAIT_FOR_RECORD }
                12,         { WRITE_BEHIND }

{ DATE PRIMARY }

                6,          { BACKUP$, }
                8,          { CREATION$, }
                10,         { EXPIRATION$, }
                8,          { REVISION$, }

{ FILE PRIMARY }

                10,         { ALLOCATION, }
                19,         { BEST_TRY_CONTIGUOUS, }
                11,         { BUCKET_SIZE, }
                12,         { CLUSTER_SIZE, }
                7,          { CONTEXT$ }
                10,         { CONTIGUOUS, }
                9,          { CREATE_IF }
                12,         { DEFAULT_NAME, }
                14,         { DEFERRED_WRITE, }
                15,         { DELETE_ON_CLOSE, }
                15,         { DIRECTORY_ENTRY, }
                15,         { ERASE_ON_DELETE, }
                9,          { EXTENSION, }
                16,         { GLOBAL_BUFFER_COUNT, }
                13,         { MT_BLOCK_SIZE, }
                19,         { MT_CURRENT_POSITION, }
                10,         { MT_NOT_EOF }
                13,         { MT_PROTECTION, }
                14,         { MT_OPEN_REWIND, }
                15,         { MT_CLOSE_REWIND }
                17,         { MAX_RECORD_NUMBER, }
                16,         { MAXIMIZE_VERSION, }
```

```
                    4,          { NAME, }
                    8,          { NOBACKUP, }
                    19,         { NON_FILE_STRUCTURED }
                    17,         { OUTPUT_FILE_PARSE }
                    12,         { ORGANIZATION, }
                    5,          { OWNER, }
                    14,         { PRINT_ON_CLOSE, }
                    10,         { PROTECTION, }
                    10,         { READ_CHECK, }
                    8,          { REVISION, }
                    15,         { SEQUENTIAL_ONLY }
                    15,         { SUBMIT_ON_CLOSE, }
                    9,          { SUPERSEDE, }
                    9,          { TEMPORARY }
                    17,         { TRUNCATE_ON_CLOSE, }
                    14,         { USER_FILE_OPEN }
                    11,         { WINDOW_SIZE }
                    11,         { WRITE_CHECK, }

{ JOURNALING PRIMARY }

                    11,         { AFTER_IMAGE, }
                    10,         { AFTER_NAME }
                    11,         { AUDIT_TRAIL, }
                    10,         { AUDIT_NAME }
                    12,         { BEFORE_IMAGE, }
                    11,         { BEFORE_NAME }
                    13,         { RECOVERY_UNIT, }

{ KEY PRIMARY }

                    7,          { CHANGES, }
                    9,          { DATA_AREA, }
                    9,          { DATA_FILL, }
                    20,         { DATA_KEY_COMPRESSION, }
                    23,         { DATA_RECORD_COMPRESSION, }
                    10,         { DUPLICATES, }
                    10,         { INDEX_AREA, }
                    17,         { INDEX_COMPRESSION, }
                    10,         { INDEX_FILL, }
                    17,         { LEVEL1_INDEX_AREA, }
                    4,          { NAME$, }
                    8,          { NULL_KEY, }
                    10,         { NULL_VALUE, }
                    6,          { PROLOG(UE) - 1ST 6 CHARS ONLY }
                    0,          { SEG_LENGTH, }
                    0,          { SEG_POSITION, }
                    0,          { SEG_TYPE, }

{ RECORD PRIMARY }

                    10,         { BLOCK_SPAN, }
                    16,         { CARRIAGE_CONTROL, }
                    18,         { CONTROL_FIELD_SIZE, }
                    6,          { FORMAT, }
                    4,          { SIZE, }
```

```
{ SHARING PRIMARY }
                    6,          { DELETE }
                    3,          { GET }
                    11,         { MULTISTREAM }
                    8,          { PROHIBIT }
                    3,          { PUT }
                    6,          { UPDATE }
                    14,         { USER_INTERLOCK }
{ SYSTEM PRIMARY }
                    6,          { DEVICE, }
                    6,          { SOURCE, }
                    6           { TARGET, }
                    );
    { +
    These are the maximum values of number-valued secondaries.
    - }
    SECONDARY_MAX := (
{ RESERVE 0 }       0,          { DUMMY_SECONDARY$, }
{ ACCESS PRIMARY }
                    0,          { BLOCK_IO$ }
                    0,          { DELETE$ }
                    0,          { GET$ }
                    0,          { PUT$ }
                    0,          { RECORD_IO$ }
                    0,          { TRUNCATE$ }
                    0,          { UPDATE$ }
{ ACL PRIMARY }
                    0,          { ENTRY }
{ ANALYSIS_OF_AREA PRIMARY }
                    0,          { RECLAIMED_SPACE }
{ ANALYSIS_OF_KEY PRIMARY }
                    0,          { DATA_FILL$, }
                    0,          { DATA_KEY_COMPRESSION, }
                    0,          { DATA_RECORD_COMPRESSION, }
                    0,          { DATA_RECORD_COUNT, }
                    0,          { DATA_SPACE_OCCUPIED, }
                    0,          { DELETIONS, }
                    0,          { DEPTH, }
                    0,          { DUPLICATES_PER_SIDR, }
                    0,          { INDEX_COMPRESSION, }
                    0,          { INDEX_FILL$, }
```

```
                    0,          { INDEX_SPACE_OCCUPIED, }
                    0,          { LEVEL1_RECORD_COUNT }
                    0,          { MEAN_DATA_LENGTH, }
                    0,          { MEAN_INDEX_LENGTH, }
                    0,          { RANDOM_ACCESSES, }
                    0,          { RANDOM_INSERTS, }
                    0,          { SEQUENTIAL_ACCESSES, }

{ AREA PRIMARY }

                    EDF$C_1GIGA,{ ALLOCATION$, }
                    0,          { BEST_TRY_CONTIGUOUS$, }
                    BKT$C_MAXBKTSIZ,    { BUCKET_SIZE$, }
                    0,          { CONTIGUOUS$, }
                    0,          { EXACT_POSITIONING$, }
                    EDF$C_1GIGA,{ EXTENSION$, }
                    16777215,   { POSITION$, }
                    65535,      { VOLUME$, }

{ CONNECT PRIMARY }

                    0,          { ASYNCHRONOUS }
                    0,          { BLOCK_IO }
                    EDF$C_1GIGA, { BUCKET_CODE }
                    EDF$C_1GIGA, { CONTEXT }
                    0,          { END_OF_FILE }
                    0,          { FILL_BUCKETS }
                    0,          { FAST_DELETE }
                    255,        { KEY_OF_REFERENCE }
                    0,          { KEY_GREATER_EQUAL }
                    0,          { KEY_GREATER_THAN }
                    0,          { KEY_LIMIT }
                    0,          { LOCATE_MODE }
                    0,          { LOCK_ON_READ }
                    0,          { LOCK_ON_WRITE }
                    0,          { MANUAL_UNLOCKING }
                    255,        { MULTIBLOCK_COUNT }
                    255,        { MULTIBUFFER_COUNT }
                    0,          { NOLOCK }
                    0,          { NONEXISTENT_RECORD }
                    0,          { READ_AHEAD }
                    0,          { READ_REGARDLESS }
                    0,          { TIMEOUT_ENABLE }
                    255,        { TIMEOUT_PERIOD }
                    0,          { TRUNCATE_ON_PUT }
                    0,          { TT_CANCEL_CONTROL_O }
                    0,          { TT_UPCASE_INPUT }
                    0,          { TT_PROMPT }
                    0,          { TT_PURGE_TYPE_AHEAD }
                    0,          { TT_READ_NOECHO }
                    0,          { TT_READ_NOFILTER }
                    0,          { UPDATE_IF }
                    0,          { WAIT_FOR_RECORD }
                    0,          { WRITE_BEHIND }

{ DATE PRIMARY }
```

```
                    0,              { BACKUP$, }
                    0,              { CREATION$, }
                    0,              { EXPIRATION$, }
                    0,              { REVISION$, }

{ FILE PRIMARY }

                    EDF$C_1GIGA,    { ALLOCATION, }
                    0,              { BEST_TRY_CONTIGUOUS, }
                    BKT$C_MAXBKTSIZ,    { BUCKET_SIZE, }
                    EDF$C_1GIGA,    { CLUSTER_SIZE, }
                    EDF$C_1GIGA,    { CONTEXT }
                    0,              { CONTIGUOUS, }
                    0,              { CREATE_IF }
                    0,              { DEFAULT_NAME, }
                    0,              { DEFERRED_WRITE, }
                    0,              { DELETE_ON_CLOSE, }
                    0,              { DIRECTORY_ENTRY, }
                    0,              { ERASE_ON_DELETE, }
                    EDF$C_1GIGA,    { EXTENSION, }
                    EDF$C_MAX_GBL_BUFS, { GLOBAL_BUFFER_COUNT, }
                    65532,          { MT_BLOCK_SIZE, }
                    0,              { MT_CURRENT_POSITION, }
                    0,              { MT_NOT_EOF }
                    0,              { MT_PROTECTION, }
                    0,              { MT_OPEN_REWIND, }
                    0,              { MT_CLOSE_REWIND }
                    EDF$C_1GIGA,    { MAX_RECORD_NUMBER, }
                    0,              { MAXIMIZE_VERSION, }
                    0,              { NAME, }
                    0,              { NOBACKUP, }
                    0,              { NON_FILE_STRUCTURED }
                    0,              { OUTPUT_FILE_PARSE }
                    0,              { ORGANIZATION, }
                    0,              { OWNER, }
                    0,              { PRINT_ON_CLOSE, }
                    0,              { PROTECTION, }
                    0,              { READ_CHECK, }
                    65535,          { REVISION, }
                    0,              { SEQUENTIAL_ONLY }
                    0,              { SUBMIT_ON_CLOSE, }
                    0,              { SUPERSEDE, }
                    0,              { TEMPORARY }
                    0,              { TRUNCATE_ON_CLOSE, }
                    0,              { USER_FILE_OPEN }
                    EDF$C_1GIGA,    { WINDOW_SIZE }
                    0,              { WRITE_CHECK, }

{ JOURNALING PRIMARY }

                    0,              { AFTER_IMAGE, }
                    0,              { AFTER_NAME }
                    0,              { AUDIT_TRAIL, }
                    0,              { AUDIT_NAME }
                    0,              { BEFORE_IMAGE, }
```

```
                    0,          { BEFORE_NAME }
                    0,          { RECOVERY_UNIT, }
{ KEY PRIMARY }

                    0,          { CHANGES, }
                    254,        { DATA_AREA, }
                    100,        { DATA_FILL, }
                    99,         { DATA_KEY_COMPRESSION, }
                    99,         { DATA_RECORD_COMPRESSION, }
                    0,          { DUPLICATES, }
                    254,        { INDEX_AREA, }
                    99,         { INDEX_COMPRESSION, }
                    100,        { INDEX_FILL, }
                    254,        { LEVEL1_INDEX_AREA, }
                    0,          { NAME$, }
                    0,          { NULL_KEY, }
                    255,        { NULL_VALUE, }
                    3,          { PROLOGUE }
                    255,        { SEG_LENGTH, }
                    16299,      { SEG_POSITION, }
                    0,          { SEG_TYPE, }
{ RECORD PRIMARY }

                    0,          { BLOCK_SPAN, }
                    0,          { CARRIAGE_CONTROL, }
                    255,        { CONTROL_FIELD_SIZE, }
                    0,          { FORMAT, }
          EDF$K_MAXRECSIZ,      { SIZE, }
{ SHARING PRIMARY }

                    0,          { DELETE }
                    0,          { GET }
                    0,          { MULTISTREAM }
                    0,          { PROHIBIT }
                    0,          { PUT }
                    0,          { UPDATE }
                    0,          { USER_INTERLOCK }
{ SYSTEM PRIMARY }

                    0,          { DEVICE, }
                    0,          { SOURCE, }
                    0           { TARGET, }

                    );

    { +
    These are the secondary value types.
     - }

    SEC_TYPE := (

{ +
```

```
        KEY:            STR,  NUM,  QUAL,  SW
- }
{ RESERVE 0 }       (FALSE,FALSE,FALSE,FALSE),          { DUMMY_SECONDARY$, }
{ ACCESS PRIMARY }
                    (FALSE,FALSE,FALSE,TRUE),           { BLOCK_IO$ }
                    (FALSE,FALSE,FALSE,TRUE),           { DELETE$ }
                    (FALSE,FALSE,FALSE,TRUE),           { GET$ }
                    (FALSE,FALSE,FALSE,TRUE),           { PUT$ }
                    (FALSE,FALSE,FALSE,TRUE),           { RECORD_IO$ }
                    (FALSE,FALSE,FALSE,TRUE),           { TRUNCATE$ }
                    (FALSE,FALSE,FALSE,TRUE),           { UPDATE$ }
{ ACL PRIMARY }
                    (TRUE,FALSE,FALSE,FALSE),           { ENTRY }
{ ANALYSIS_OF_AREA PRIMARY }
                    (FALSE,FALSE,FALSE,FALSE),          { RECLAIMED_SPACE }
{ ANALYSIS_OF_KEY PRIMARY }
                    (FALSE,FALSE,FALSE,FALSE),          { DATA_FILL$, }
                    (FALSE,FALSE,FALSE,FALSE),          { DATA_KEY_COMPRESSION, }
                    (FALSE,FALSE,FALSE,FALSE),          { DATA_RECORD_COMPRESSION, }
                    (FALSE,FALSE,FALSE,FALSE),          { DATA_RECORD_COUNT, }
                    (FALSE,FALSE,FALSE,FALSE),          { DATA_SPACE_OCCUPIED, }
                    (FALSE,FALSE,FALSE,FALSE),          { DELETIONS, }
                    (FALSE,FALSE,FALSE,FALSE),          { DEPTH, }
                    (FALSE,FALSE,FALSE,FALSE),          { DUPLICATES_PER_SIDR, }
                    (FALSE,FALSE,FALSE,FALSE),          { INDEX_COMPRESSION, }
                    (FALSE,FALSE,FALSE,FALSE),          { INDEX_FILL$, }
                    (FALSE,FALSE,FALSE,FALSE),          { INDEX_SPACE_OCCUPIED, }
                    (FALSE,FALSE,FALSE,FALSE),          { LEVEL1_RECORD_COUNT }
                    (FALSE,FALSE,FALSE,FALSE),          { MEAN_DATA_LENGTH, }
                    (FALSE,FALSE,FALSE,FALSE),          { MEAN_INDEX_LENGTH, }
                    (FALSE,FALSE,FALSE,FALSE),          { RANDOM_ACCESSES, }
                    (FALSE,FALSE,FALSE,FALSE),          { RANDOM_INSERTS, }
                    (FALSE,FALSE,FALSE,FALSE),          { SEQUENTIAL_ACCESSES, }
{ AREA PRIMARY }
                    (FALSE,TRUE,FALSE,FALSE),           { ALLOCATION$, }
                    (FALSE,FALSE,FALSE,TRUE),           { BEST_TRY_CONTIGUOUS$, }
                    (FALSE,TRUE,FALSE,FALSE),           { BUCKET_SIZE$, }
                    (FALSE,FALSE,FALSE,TRUE),           { CONTIGUOUS$, }
                    (FALSE,FALSE,FALSE,TRUE),           { EXACT_POSITIONING$, }
                    (FALSE,TRUE,FALSE,FALSE),           { EXTENSION$, }
                    (FALSE,FALSE,FALSE,FALSE),          { POSITION$, }
                    (FALSE,TRUE,FALSE,FALSE),           { VOLUME$, }
{ +
        KEY:            STR,  NUM,  QUAL,  SW
```

```
- }
{ CONNECT PRIMARY }

                    (FALSE,FALSE,FALSE,TRUE),       { ASYNCHRONOUS }
                    (FALSE,FALSE,FALSE,TRUE),       { BLOCK_IO }
                    (FALSE,TRUE,FALSE,FALSE),       { BUCKET_CODE }
                    (FALSE,TRUE,FALSE,FALSE),       { CONTEXT }
                    (FALSE,FALSE,FALSE,TRUE),       { END_OF_FILE }
                    (FALSE,FALSE,FALSE,TRUE),       { FILL_BUCKETS }
                    (FALSE,FALSE,FALSE,TRUE),       { FAST_DELETE }
                    (FALSE,TRUE,FALSE,FALSE),       { KEY_OF_REFERENCE }
                    (FALSE,FALSE,FALSE,TRUE),       { KEY_GREATER_EQUAL }
                    (FALSE,FALSE,FALSE,TRUE),       { KEY_GREATER_THAN }
                    (FALSE,FALSE,FALSE,TRUE),       { KEY_LIMIT }
                    (FALSE,FALSE,FALSE,TRUE),       { LOCATE_MODE }
                    (FALSE,FALSE,FALSE,TRUE),       { LOCK_ON_READ }
                    (FALSE,FALSE,FALSE,TRUE),       { LOCK_ON_WRITE }
                    (FALSE,FALSE,FALSE,TRUE),       { MANUAL_UNLOCKING }
                    (FALSE,TRUE,FALSE,FALSE),       { MULTIBLOCK_COUNT }
                    (FALSE,TRUE,FALSE,FALSE),       { MULTIBUFFER_COUNT }
                    (FALSE,FALSE,FALSE,TRUE),       { NOLOCK }
                    (FALSE,FALSE,FALSE,TRUE),       { NONEXISTENT_RECORD }
                    (FALSE,FALSE,FALSE,TRUE),       { READ_AHEAD }
                    (FALSE,FALSE,FALSE,TRUE),       { READ_REGARDLESS }
                    (FALSE,FALSE,FALSE,TRUE),       { TIMEOUT_ENABLE }
                    (FALSE,TRUE,FALSE,FALSE),       { TIMEOUT_PERIOD }
                    (FALSE,FALSE,FALSE,TRUE),       { TRUNCATE_ON_PUT }
                    (FALSE,FALSE,FALSE,TRUE),       { TT_CANCEL_CONTROL_O }
                    (FALSE,FALSE,FALSE,TRUE),       { TT_UPCASE_INPUT }
                    (FALSE,FALSE,FALSE,TRUE),       { TT_PROMPT }
                    (FALSE,FALSE,FALSE,TRUE),       { TT_PURGE_TYPE_AHEAD }
                    (FALSE,FALSE,FALSE,TRUE),       { TT_READ_NOECHO }
                    (FALSE,FALSE,FALSE,TRUE),       { TT_READ_NOFILTER }
                    (FALSE,FALSE,FALSE,TRUE),       { UPDATE_IF }
                    (FALSE,FALSE,FALSE,TRUE),       { WAIT_FOR_RECORD }
                    (FALSE,FALSE,FALSE,TRUE),       { WRITE_BEHIND }

{ DATE PRIMARY }

                    (TRUE,FALSE,FALSE,FALSE),       { BACKUP$, }
                    (TRUE,FALSE,FALSE,FALSE),       { CREATION$, }
                    (TRUE,FALSE,FALSE,FALSE),       { EXPIRATION$, }
                    (TRUE,FALSE,FALSE,FALSE),       { REVISION$, }

{ FILE PRIMARY }

                    (FALSE,TRUE,FALSE,FALSE),       { ALLOCATION, }
                    (FALSE,FALSE,FALSE,TRUE),       { BEST_TRY_CONTIGUOUS, }
                    (FALSE,TRUE,FALSE,FALSE),       { BUCKET_SIZE, }
                    (FALSE,TRUE,FALSE,FALSE),       { CLUSTER_SIZE, }
                    (FALSE,TRUE,FALSE,FALSE),       { CONTEXT$ }
                    (FALSE,FALSE,FALSE,TRUE),       { CONTIGUOUS, }
                    (FALSE,FALSE,FALSE,TRUE),       { CREATE_IF }
                    (TRUE,FALSE,FALSE,FALSE),       { DEFAULT_NAME, }
                    (FALSE,FALSE,FALSE,TRUE),       { DEFERRED_WRITE, }
```

```
                        (FALSE,FALSE,FALSE,TRUE),       { DELETE_ON_CLOSE, }
                        (FALSE,FALSE,FALSE,TRUE),       { DIRECTORY_ENTRY, }
                        (FALSE,FALSE,FALSE,TRUE),       { ERASE_ON_DELETE, }
                        (FALSE,TRUE,FALSE,FALSE),       { EXTENSION, }
                        (FALSE,TRUE,FALSE,FALSE),       { GLOBAL_BUFFER_COUNT, }
                        (FALSE,TRUE,FALSE,FALSE),       { MT_BLOCK_SIZE, }
                        (FALSE,FALSE,FALSE,TRUE),       { MT_CURRENT_POSITION, }
                        (FALSE,FALSE,FALSE,TRUE),       { MT_NOT_EOF }
                        (FALSE,FALSE,FALSE,FALSE),      { MT_PROTECTION, }
                        (FALSE,FALSE,FALSE,TRUE),       { MT_OPEN_REWIND, }
                        (FALSE,FALSE,FALSE,TRUE),       { MT_CLOSE_REWIND }
                        (FALSE,TRUE,FALSE,FALSE),       { MAX_RECORD_NUMBER, }
                        (FALSE,FALSE,FALSE,TRUE),       { MAXIMIZE_VERSION, }
                        (TRUE,FALSE,FALSE,FALSE),       { NAME, }
                        (FALSE,FALSE,FALSE,TRUE),       { NOBACKUP, }
                        (FALSE,FALSE,FALSE,TRUE),       { NON_FILE_STRUCTURED }
                        (FALSE,FALSE,FALSE,TRUE),       { OUTPUT_FILE_PARSE }
                        (FALSE,FALSE,TRUE,FALSE),       { ORGANIZATION, }
                        (FALSE,FALSE,FALSE,FALSE),      { OWNER, }
                        (FALSE,FALSE,FALSE,TRUE),       { PRINT_ON_CLOSE, }
                        (FALSE,FALSE,FALSE,FALSE),      { PROTECTION, }
                        (FALSE,FALSE,FALSE,TRUE),       { READ_CHECK, }
                        (FALSE,TRUE,FALSE,FALSE),       { REVISION, }
                        (FALSE,FALSE,FALSE,TRUE),       { SEQUENTIAL_ONLY }
                        (FALSE,FALSE,FALSE,TRUE),       { SUBMIT_ON_CLOSE, }
                        (FALSE,FALSE,FALSE,TRUE),       { SUPERSEDE, }
                        (FALSE,FALSE,FALSE,TRUE),       { TEMPORARY }
                        (FALSE,FALSE,FALSE,TRUE),       { TRUNCATE_ON_CLOSE, }
                        (FALSE,FALSE,FALSE,TRUE),       { USER_FILE_OPEN }
                        (FALSE,TRUE,FALSE,FALSE),       { WINDOW_SIZE }
                        (FALSE,FALSE,FALSE,TRUE),       { WRITE_CHECK, }

{ +
        KEY:            STR,  NUM,  QUAL,  SW
- }

{ JOURNAL PRIMARY }

                        (FALSE,FALSE,FALSE,TRUE),       { AFTER_IMAGE, }
                        (TRUE,FALSE,FALSE,FALSE),       { AFTER_NAME }
                        (FALSE,FALSE,FALSE,TRUE),       { AUDIT_TRAIL, }
                        (TRUE,FALSE,FALSE,FALSE),       { AUDIT_NAME }
                        (FALSE,FALSE,FALSE,TRUE),       { BEFORE_IMAGE, }
                        (TRUE,FALSE,FALSE,FALSE),       { BEFORE_NAME }
                        (FALSE,FALSE,TRUE,FALSE),       { RECOVERY_UNIT, }

{ KEY PRIMARY }

                        (FALSE,FALSE,FALSE,TRUE),       { CHANGES, }
                        (FALSE,TRUE,FALSE,FALSE),       { DATA_AREA, }
                        (FALSE,TRUE,FALSE,FALSE),       { DATA_FILL, }
                        (FALSE,FALSE,FALSE,TRUE),       { DATA_KEY_COMPRESSION, }
                        (FALSE,FALSE,FALSE,TRUE),       { DATA_RECORD_COMPRESSION, }
                        (FALSE,FALSE,FALSE,TRUE),       { DUPLICATES, }
                        (FALSE,TRUE,FALSE,FALSE),       { INDEX_AREA, }
                        (FALSE,FALSE,FALSE,TRUE),       { INDEX_COMPRESSION, }
```

```
                    (FALSE,TRUE,FALSE,FALSE),           { INDEX_FILL, }
                    (FALSE,TRUE,FALSE,FALSE),           { LEVEL1_INDEX_AREA, }
                    (TRUE,FALSE,FALSE,FALSE),           { NAME$, }
                    (FALSE,FALSE,FALSE,TRUE),           { NULL_KEY, }
                    (FALSE,FALSE,FALSE,FALSE),          { NULL_VALUE, }
                    (FALSE,TRUE,FALSE,FALSE),           { PROLOGUE }
                    (FALSE,TRUE,FALSE,FALSE),           { SEG_LENGTH, }
                    (FALSE,TRUE,FALSE,FALSE),           { SEG_POSITION, }
                    (FALSE,FALSE,TRUE,FALSE),           { SEG_TYPE, }
{ RECORD PRIMARY }
                    (FALSE,FALSE,FALSE,TRUE),           { BLOCK_SPAN, }
                    (FALSE,FALSE,TRUE,FALSE),           { CARRIAGE_CONTROL, }
                    (FALSE,TRUE,FALSE,FALSE),           { CONTROL_FIELD_SIZE, }
                    (FALSE,FALSE,TRUE,FALSE),           { FORMAT, }
                    (FALSE,TRUE,FALSE,FALSE),           { SIZE, }
{ +
        KEY:          STR,  NUM,  QUAL,  SW
- }
{ SHARING PRIMARY }
                    (FALSE,FALSE,FALSE,TRUE),           { DELETE }
                    (FALSE,FALSE,FALSE,TRUE),           { GET }
                    (FALSE,FALSE,FALSE,TRUE),           { MULTISTREAM }
                    (FALSE,FALSE,FALSE,TRUE),           { PROHIBIT }
                    (FALSE,FALSE,FALSE,TRUE),           { PUT }
                    (FALSE,FALSE,FALSE,TRUE),           { UPDATE }
                    (FALSE,FALSE,FALSE,TRUE),           { USER_INTERLOCK }
{ SYSTEM PRIMARY }
                    (TRUE,FALSE,FALSE,FALSE),           { DEVICE, }
                    (FALSE,FALSE,TRUE,FALSE),           { SOURCE, }
                    (FALSE,FALSE,TRUE,FALSE)            { TARGET, }

                    );
```

```
{ +
This is the QTAB array, which controls the asking and processing of questions.
- }
QTAB    :=      (

{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_DATA_FILE_NAME }
(STRING_ANSWER,         TRUE,           0,              0,              0,              0,              0),
{ EDF$K_FDL_TITLE }
(STRING_ANSWER,         TRUE,           0,              0,              0,              0,              0),
{ EDF$K_KEY_NAME }
(STRING_ANSWER,         TRUE,           0,              0,              0,              0,              0),
{ EDF$K_ANALYSIS }
(STRING_ANSWER,         TRUE,           0,              0,              0,              0,              0),
{ EDF$K_OUTPUT }
(STRING_ANSWER,         TRUE,           0,              0,              0,              0,              0),
{ EDF$K_DATA_KEY_COMP }
(REAL_ANSWER,           TRUE,           0,              -99,            99,             0,              0),
{ EDF$K_DATA_RECORD_COMP }
(REAL_ANSWER,           TRUE,           0,              -99,            99,             0,              0),
{ EDF$K_INDEX_RECORD_COMP }
(REAL_ANSWER,           TRUE,           0,              -99,            99,             0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_KEY_COMP_WANTED }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_YES,      0,              0,              0,              0),
{ EDF$K_REC_COMP_WANTED }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_YES,      0,              0,              0,              0),
{ EDF$K_IDX_COMP_WANTED }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_YES,      0,              0,              0,              0),
{ EDF$K_ASCENDING_ADDED }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_NO,       0,              0,              0,              0),
{ EDF$K_ASCENDING_LOAD }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_NO,       0,              0,              0,              0),
{ EDF$K_BLOCK_SPAN }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_YES,      0,              0,              0,              0),
{ EDF$K_CONFIRM }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_NO,       0,              0,              0,              0),
{ EDF$K_SEGMENTED }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_NO,       0,              0,              0,              0),
{ EDF$K_GLOBAL_WANTED }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_NO,       0,              0,              0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_KEY_CHANGES }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_YES,      0,              0,              0,              0),
{ EDF$K_KEY_DIST }
(BOOLEAN_ANSWER,        TRUE,           EDF$K_NO,       0,              0,              0,              0),
{ EDF$K_KEY_DUPS }
```

```
(BOOLEAN_ANSWER,        TRUE,           EDF$K_NO,       0,              0,              0,              0),
{ EDF$K_RETURN }
(BOOLEAN_ANSWER,        TRUE,           0,              0,              0,              0,              0),
{ EDF$K_CLUSTER_SIZE }
(INTEGER_ANSWER,        TRUE,           3,              1,              EDF$C_1GIGA,    0,              0),
{ EDF$K_ACTIVE_KEY }
(INTEGER_ANSWER,        TRUE,           0,              0,              0,              0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_ADDED_COUNT }
(INTEGER_ANSWER,        TRUE,           0,              0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_ADDED_COUNT_LOW }
(INTEGER_ANSWER,        TRUE,           0,              0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_ADDED_COUNT_HIGH }
(INTEGER_ANSWER,        TRUE,           100000,         0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_BLOCKS_IN_BUCKET }
(INTEGER_ANSWER,        TRUE,           32,             1,              BKT$C_MAXBKTSIZ, 0,             0),
{ EDF$K_BUCKET_WEIGHT }
(KEYWORD_ANSWER,        TRUE,   EDF$K_FLATTER_FILES,    0,              0,              0,              0),
{ EDF$K_CARR_CTRL }
(KEYWORD_ANSWER,        TRUE,           FDL$C_CR,       0,              0,              0,              0),
{ EDF$K_CONTROL_SIZE }
(INTEGER_ANSWER,        TRUE,           2,              1,              255,            0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_CURRENT_FUNCTION }
(KEYWORD_ANSWER,        TRUE,           EDF$K_HELP,     0,              0,              0,              0),
{ EDF$K_DESIGN_CYCLE }
(KEYWORD_ANSWER,        TRUE,           EDF$K_WP,       0,              0,              0,              0),
{ EDF$K_DESIRED_FILL }
(INTEGER_ANSWER,        TRUE,           100,            0,              100,            0,              0),
{ EDF$K_FILL_LOW }
(INTEGER_ANSWER,        TRUE,           50,             0,              100,            0,              0),
{ EDF$K_FILL_HIGH }
(INTEGER_ANSWER,        TRUE,           100,            0,              100,            0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_GLOBAL_COUNT }
(INTEGER_ANSWER,        FALSE,          0,              0,              65535,          0,              0),
{ EDF$K_GRANULARITY }
(KEYWORD_ANSWER,        TRUE,           EDF$K_THREE,    0,              0,              0,              0),
{ EDF$K_INITIAL_COUNT }
(INTEGER_ANSWER,        FALSE,          0,              0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_INITIAL_COUNT_LOW }
(INTEGER_ANSWER,        TRUE,           0,              0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_INITIAL_COUNT_HIGH }
(INTEGER_ANSWER,        TRUE,           100000,         0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_KEY_POSITION }
(INTEGER_ANSWER,        TRUE,           0,              0,          EDF$K_MAXRECSIZ,    0,              0),
{ EDF$K_KEY_LOW }
```

```
(INTEGER_ANSWER,        TRUE,           1,              0,              0,              0,              0),
{ EDF$K_KEY_HIGH }
(INTEGER_ANSWER,        TRUE,           255,            0,              0,              0,              0),
{ EDF$K_KEY_SIZE }
(INTEGER_ANSWER,        FALSE,          0,              0,              0,              0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_KEY_TYPE }
(KEYWORD_ANSWER,        TRUE,           FDL$C_STG,              0,              0,              0,              0),
{ EDF$K_LOAD_METHOD }
(KEYWORD_ANSWER,        TRUE,   EDF$K_FAST_CONVERT,     0,              0,              0,              0),
{ EDF$K_MAX_RECORD_SIZE }
(INTEGER_ANSWER,        FALSE,          0,              0,              0,              0,              0),
{ EDF$K_MEAN_RECORD_SIZE }
(INTEGER_ANSWER,        FALSE,          0,              1,      EDF$K_MAXRECSIZ,        0,              0),
{ EDF$K_NUMBER_DUPS }
(INTEGER_ANSWER,        TRUE,           0,              0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_NUMBER_KEYS }
(INTEGER_ANSWER,        TRUE,           1,              1,              255,            0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_NUMBER_RECORDS }
(INTEGER_ANSWER,        FALSE,          0,              0,              EDF$C_1GIGA,    0,              0),
{ EDF$K_PROLOGUE_VERSION }
(INTEGER_ANSWER,        TRUE,           3,              0,              3,              0,              0),
{ EDF$K_PROMPTING }
(KEYWORD_ANSWER,        TRUE,           EDF$K_FULL,     0,              0,              0,              0),
{ EDF$K_RECORD_FORMAT }
(KEYWORD_ANSWER,        TRUE,           FDL$C_VAR,      0,              0,              0,              0),
{ EDF$K_RESPONSES }
(KEYWORD_ANSWER,        TRUE,           EDF$K_AUTO,     0,              0,              0,              0),
{ EDF$K_SCRIPT_OPTION }
(KEYWORD_ANSWER,        FALSE,          0,              0,              0,              0,              0),
{ EDF$K_SET_FUNCTION }
(KEYWORD_ANSWER,        FALSE,          0,              0,              0,              0,              0),
{ EDF$K_SIZE_LOW }
(INTEGER_ANSWER,        TRUE,           1,              1,      EDF$K_MAXRECSIZ,        0,              0),
{ EDF$K_SIZE_HIGH }
(INTEGER_ANSWER,        TRUE,           1000,           1,      EDF$K_MAXRECSIZ,        0,              0),
{ EDF$K_SURFACE_OPTION }
(KEYWORD_ANSWER,        TRUE,   EDF$K_LINE_SURFACE,     0,              0,              0,              0),
{ +
QUESTION_OFFSET
ANSWER_CLASS,           DEFAULT_OK,     DEFAULT,        LOW_BOUND,      HIGH_BOUND,     KEY_TABLE,      STATE_TABLE
- }
{ EDF$K_TEST_PRIMARY }
(KEYWORD_ANSWER,        TRUE,       FDL$C_FILE,         0,              0,              0,              0),
{ EDF$K_TEST_SECONDARY }
(OBJECT_ANSWER,         FALSE,          0,              0,              0,              0,              0),
{ EDF$K_TEST_SECONDARY_VALUE }
(OBJECT_ANSWER,         FALSE,          0,              0,              0,              0,              0)
```

```
                );

NULL_STRING             := (

        0,                      { DSC$W_LENGTH }
        DSC$K_DTYPE_T,          { DSC$B_DTYPE }
        DSC$K_CLASS_D,          { DSC$B_CLASS }
        NIL                     { DSC$A_POINTER }

                );

LINE_OBJECT_TEMPLATE    := (

        SEC,                    { LINE_OBJECT_TYPE }
        NIL,                    { FORE }
        NIL,                    { BACK }

        (                       { COMMENT }
        0,
        DSC$K_DTYPE_T,
        DSC$K_CLASS_D,
        NIL
        ),

        (                       { STRING }
        0,
        DSC$K_DTYPE_T,
        DSC$K_CLASS_D,
        NIL
        ),

        KEY,                    { PRIMARY }
        0,                      { PRINUM }
        DUMMY_SECONDARY$,       { SECONDARY }
        0,                      { SECNUM }
        0,                      { QUALIFIER }
        0,                      { NUMBER }
        TRUE,                   { SWITCH }
        0,                      { OWNER_UIC }

        (                       { PROT_MASK }
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
```

```
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE,
        FALSE
        ),

        0,              { FID1 }
        0,              { FID2 }
        0               { FID3 }

        );

    { End of File SRC$:EDFVALUE.PAS }
```

EDF

EDFVALUE
PAS

EDF
MAP

EDFSTRUCT
SDL

DUMPMSG
LIS

EDFASK
LIS